Controlador de Elevador

Projeto Final - MC613

Dupla (F12):
Eduardo Moreira Freitas de Souza RA: 166779
Laura Politi de Oliveira RA: 171803

Sumário

1. Introdução

O controlador de elevador pode controlar um elevador para um prédio de exatamente oito andares. Cada andar possui botões de chamada independentes para subida e descida. O monitor mostra um esquema de oito andares e seus respectivos botões de subida e descida. Os blocos maiores representam os andares, mudando de cor quando um andar é o selecionado pelo usuário no painel interno do elevador. Os blocos menores representam os botões de chamada (subida e descida) e também mudam de cor quando acionados. O elevador é representado por um retângulo em cima dos blocos de andares que muda de posição de acordo com o andar atual do elevador. Quando o elevador pára em um dos andares, ele fica um tempo estacionado, com as portas abertas, antes de continuar seu trajeto.

2. Diagrama de blocos

Segue abaixo os Diagramas de Blocos referentes à implementação desenvolvida do controlador de elevador.

2.1 controlador_geral

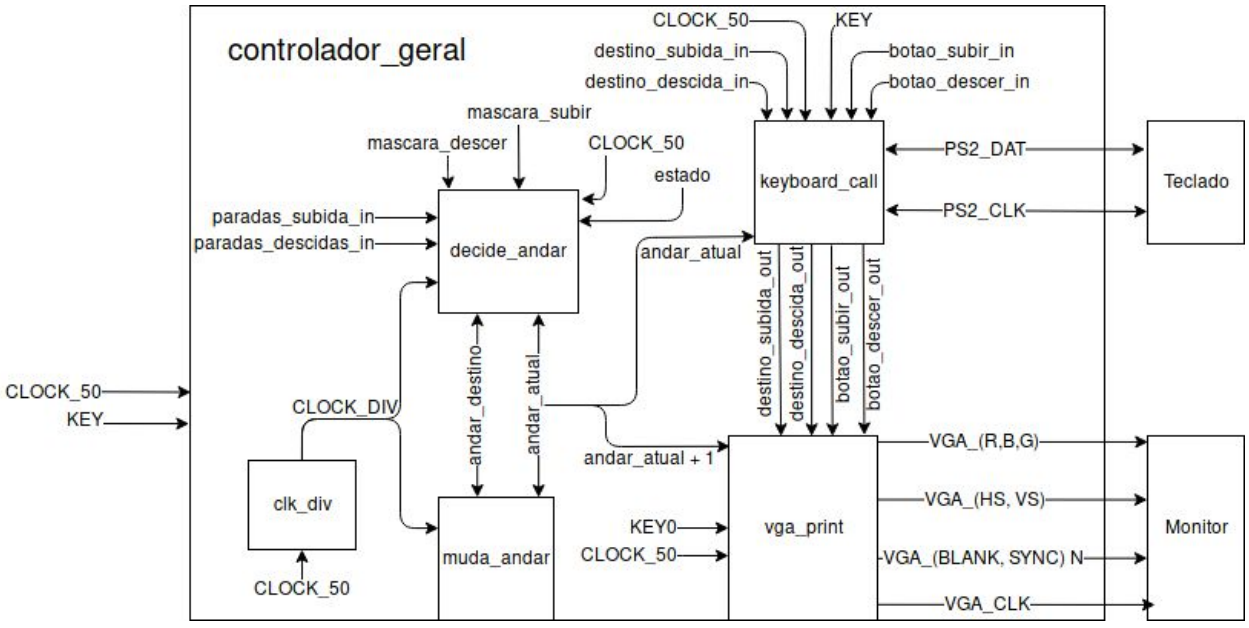

Figura 1 - Diagrama de Blocos do Controlador Geral

2.2 keyboard_call

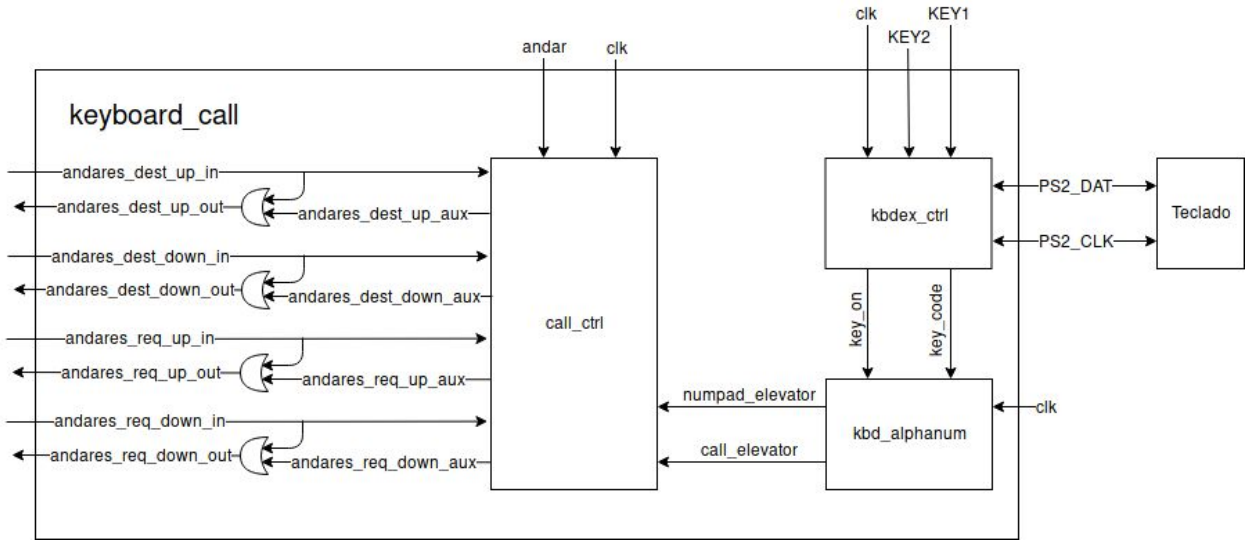

Figura 2 - Diagrama de Blocos do keyboard_call.

2.3 vga_print

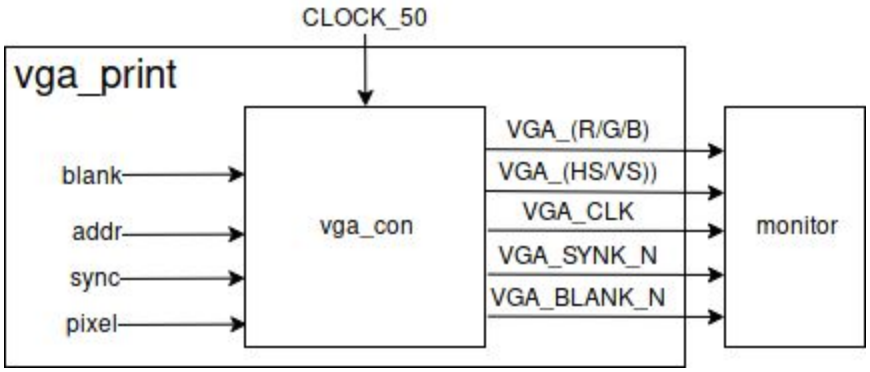

Figura 3 - Diagrama de Blocos do vga_print

3. Descrição funcional

O sistema controla o funcionamento do elevador através do Controlador Geral, que coleta as informações de sub-controladores e realiza a comunicação entre as componentes. Essas componentes controlam os andares em que o elevador é chamado e os andares de destino do elevador, e realizam seu movimento.

O elevador terá seu funcionamento baseado em dois estados: um de subida e outro de descida. Quando o elevador está no estado de subida, ele avalia qual o andar mais baixo em que o elevador foi chamado até o momento cujo destino seja algum andar acima dele (ou seja, o usuário do elevador quer subir), se movimenta até esse andar, e realiza a subida, parando nos andares que também apertaram o botão de subida ou nos andares de destino dos usuários. Quando o elevador se encontra no estado de descida, ele se move até o andar mais alto que solicitou o elevador a fim de descer, e realiza a descida do prédio de forma análoga ao estado de subida.

Quando o elevador pára em algum dos andares, ele permanece um tempo com as portas abertas para permitir a entrada e saída dos usuários. É também nesse momento que os botões de chamada ou internos que estavam ativados no andar em que se encontra o elevador são desativados.

3.1 - controlador_geral

É a entidade *"top-level"* do nosso projeto. É essa componente que realiza a comunicação entre os sinais de entrada e saída das demais componentes. Internamente ao programa, foram utilizados alguns vetores (*destino_subida, destino_descida, botao_subir* e *botao_descer*) para guardar o estado das chamadas do elevador e as variáveis *andar_atual* e *andar_destino* que definem a posição e o comportamento futuro do elevador. Além disso, a variável *estado* define se o elevador se encontra no ciclo de subida ou de descida.

3.2 - decide_andar

É a entidade que faz o controle do movimento do elevador. Ela lida com a informação sobre os andares em que o elevador deve parar, seja devido à chamada do elevador nesse andar ou ao fato do andar ser um andar de destino definido pelo usuário no painel interno do elevador. Essas informações são tratadas de forma separada para as etapas de subida e de descida.

Na etapa de subida, o elevador se move até o andar mais baixo que requisitou o movimento para cima, ou até o andar de destino mais baixo selecionado, e a partir desse andar, realiza a subida do prédio, parando nos andares de destino ou nos demais andares que desejam subir. A etapa de descida ocorre de forma análoga, com o elevador subindo até o andar mais alto que realizou uma chamada para descer e a partir dele realiza a descida.

Caso o elevador esteja parado, ou seja, nenhum botão esteja ativado, ele reveza entre os estados de subida e descida e verifica se algum novo botão foi apertado. Também é definido um tempo de 5s em que o elevador fica parado no andar com as portas abertas antes de continuar sua trajetória. Se não houver nenhuma chamada do elevador, ele fica parado no andar com as portas abertas.

3.3 - muda_andar

Essa entidade funciona como o motor do elevador. É ela que altera a posição do elevador, baseado no andar de destino definido pela entidade *decide_andar*. Ela utiliza o clock de 1Hz como base, o que faz com que o tempo de movimentação entre andares vizinhos seja de 1 segundo.

3.4 - clk_div

É um divisor de clock: gera um clock de 1Hz a partir do clock de 50MHz de entrada.

3.5 - keyboard_call

É a entidade que lida com o recebimento de chamadas nos andares e no painel interno do elevador através do teclado, e faz as ligações entre as entidades *call_ctrl* (atualiza os vetores de andares requisitados), *kbdex_ctrl* (lê o sinal serial do teclado e o transforma em um std_logic_vector que guarda o scan da tecla pressionada) e *kbd_alphanum* (interpreta qual tecla foi pressionada e informa call_ctrl sobre quais bits têm de ser ativados no vetor de andares).

3.6 - call_ctrl

Atualiza o vetor de subida/descida (*andares_req_(up/down)*) e o vetor de destinos do elevador (*andares_dest_(up/down)*) de acordo com os botões pressionados .

3.7 - kbdex_ctrl

Recebe o sinal serial do teclado (*ps2_data*), o interpreta e passa um *std_logic_vector* (*key_code*) para *kbd_alphanum* contendo o *scan code* da tecla pressionada.

3.8 - kbd_alphanum

Recebe o *scan code* (key_code) de *kbdex_ctrl*, faz a interpretação de qual tecla foi pressionada e envia para *call_ctrl* dois *std_logic_vector* de tamanho 8: *numpad_elevator*, cujo bit ativado representa a tecla pressionada de dentro do elevador, e *call_elevator*, que representa qual andar chamou o elevador e em qual sentido.

3.9 - vga_print

É a entidade que mapeia os pixels da tela para imprimir os blocos que representam os andares, os botões e o elevador. Ela engloba a entidade vgacon, que realizará a comunicação com o monitor. Os blocos foram mapeados de forma fixa na tela e a cor de cada bloco depende dos vetores de entrada que representam o estado das chamadas do elevador (*dest_subida, dest_descida, botao_subir* e *botao_descer*) e a posição atual do elevador (*andar*).

3.10 - Outros arquivos

Alguns arquivos utilizados no projeto foram os arquivos disponibilizados na pasta Material complementar do moodle da disciplina. Eles são: *kbdex_ctrl.vhd, ps2_iobase.vhd, vga_pll_0002.qip, vga_pll_0002.v, vga_pll.vhd, vgacon.vhd.* Esses arquivos foram utilizados para realizar a comunicação com os periféricos. Além disso, também foi utilizado o *pin_assignment* disponibilizado na mesma pasta.

3.11 - Considerações adicionais: Depuração

Para fins de depuração do código, apenas, o visor de 7 segmentos mostra em *HEX5* o andar de destino e em *HEX4* o andar atual do elevador. Além disso, *LEDR9* indica o estado atual do elevador. O LED está apagado no estado de subida e aceso no estado de descida. O arquivo *bin2hex.vhd* foi reutilizado dos laboratórios para mostrar as informações no visor da placa. Essas informações não fazem parte da demonstração final do projeto.

4. Execução do programa

4.1 Tela

Utilizamos o monitor para que fosse possível avaliar o comportamento do programa. Na Figura 4, reproduzimos a tela base do nosso programa. Nela é possível identificar três elementos principais. Os retângulos brancos representam os andares do prédio, sendo o da esquerda o mais baixo e o da direita o mais alto. A dupla de quadrados abaixo dos andares representam os botões de chamada do elevador em cada um dos andares, sendo o botão de cima o de subida e o de baixo o de descida. É válido observar que o primeiro andar não possui um botão de descida e o último andar não possui um botão de subida. O último elemento é o elevador, que está representado na tela como um retângulo verde sobre os andares.

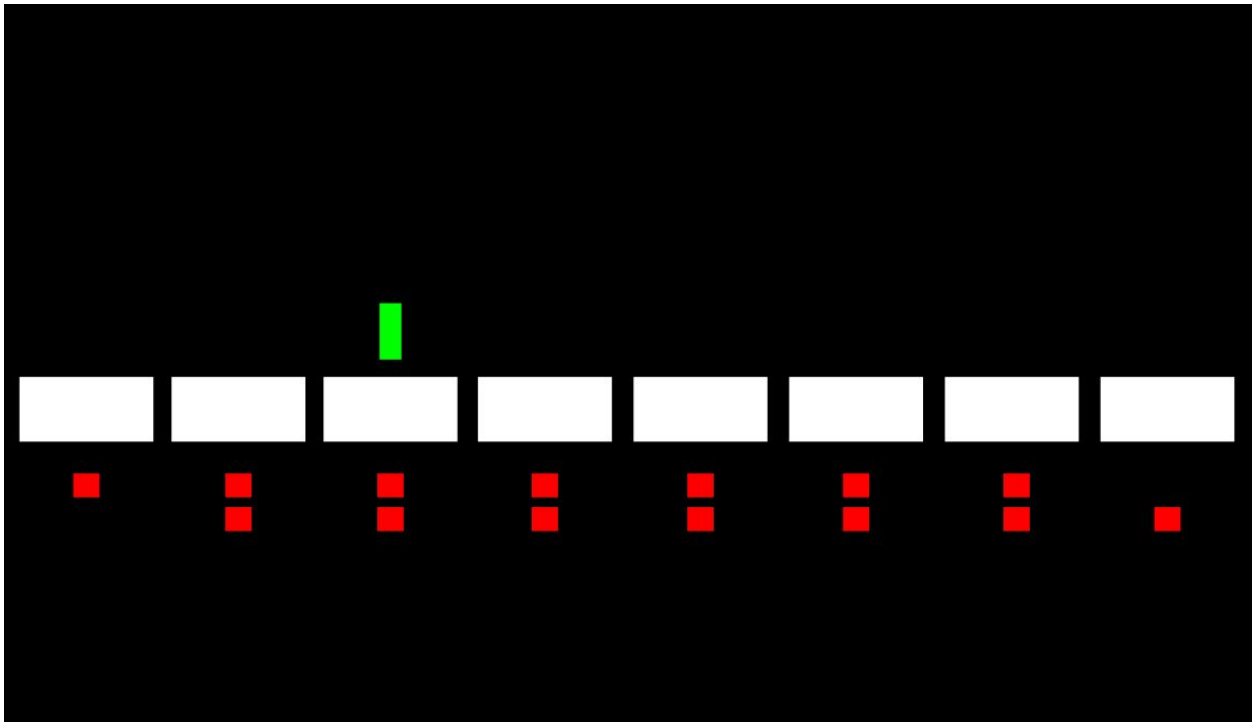

Figura 4 - Ilustração da tela base do programa.

A Figura 5 ilustra o elevador em funcionamento. O estado do sistema é determinado pelas cores dos elementos na tela. A cor dos andares indica se o andar foi selecionado no painel interno do elevador - branco significa que o andar não está selecionado e azul significa que ele está. Os botões de chamada do elevador estarão vermelhos caso estejam desativados e verdes caso ativados. O elevador é sempre verde, porém sua posição indica o andar que o elevador está naquele instante.

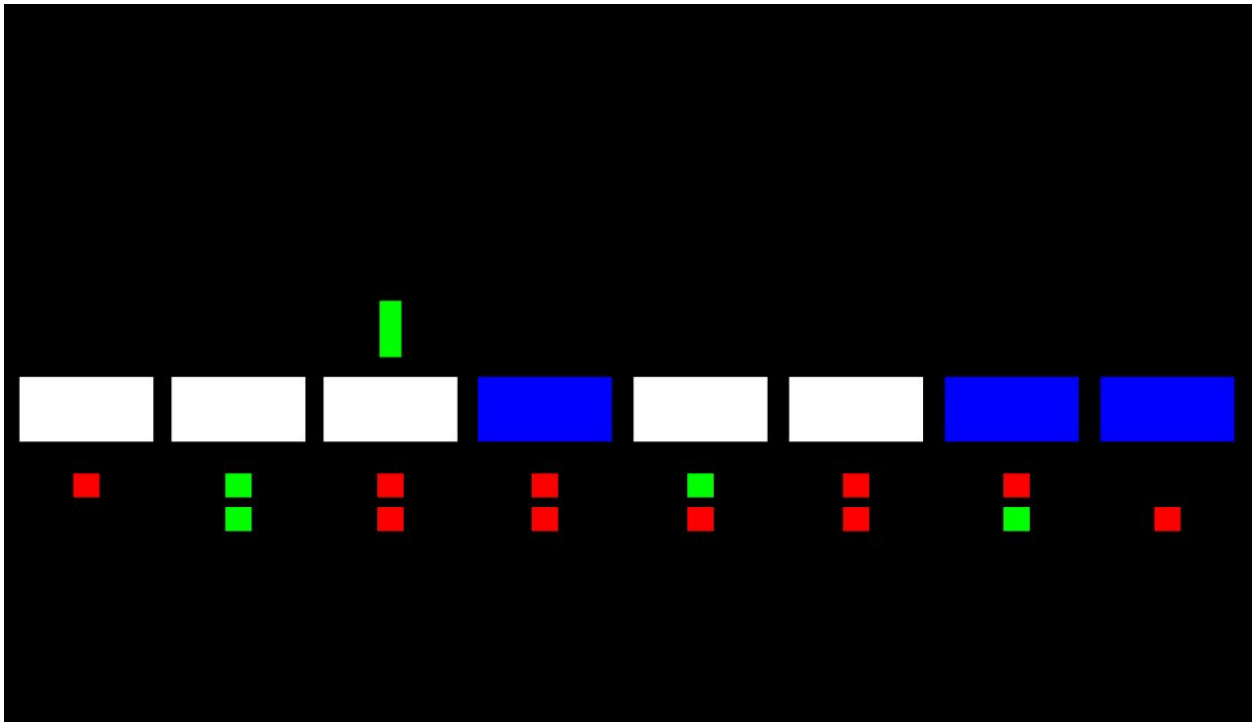

Figura 5 - Ilustração da tela durante o funcionamento do elevador.

4.2 Teclado

Segue abaixo o mapeamento dos botões de controle do elevador no teclado alfanumérico.

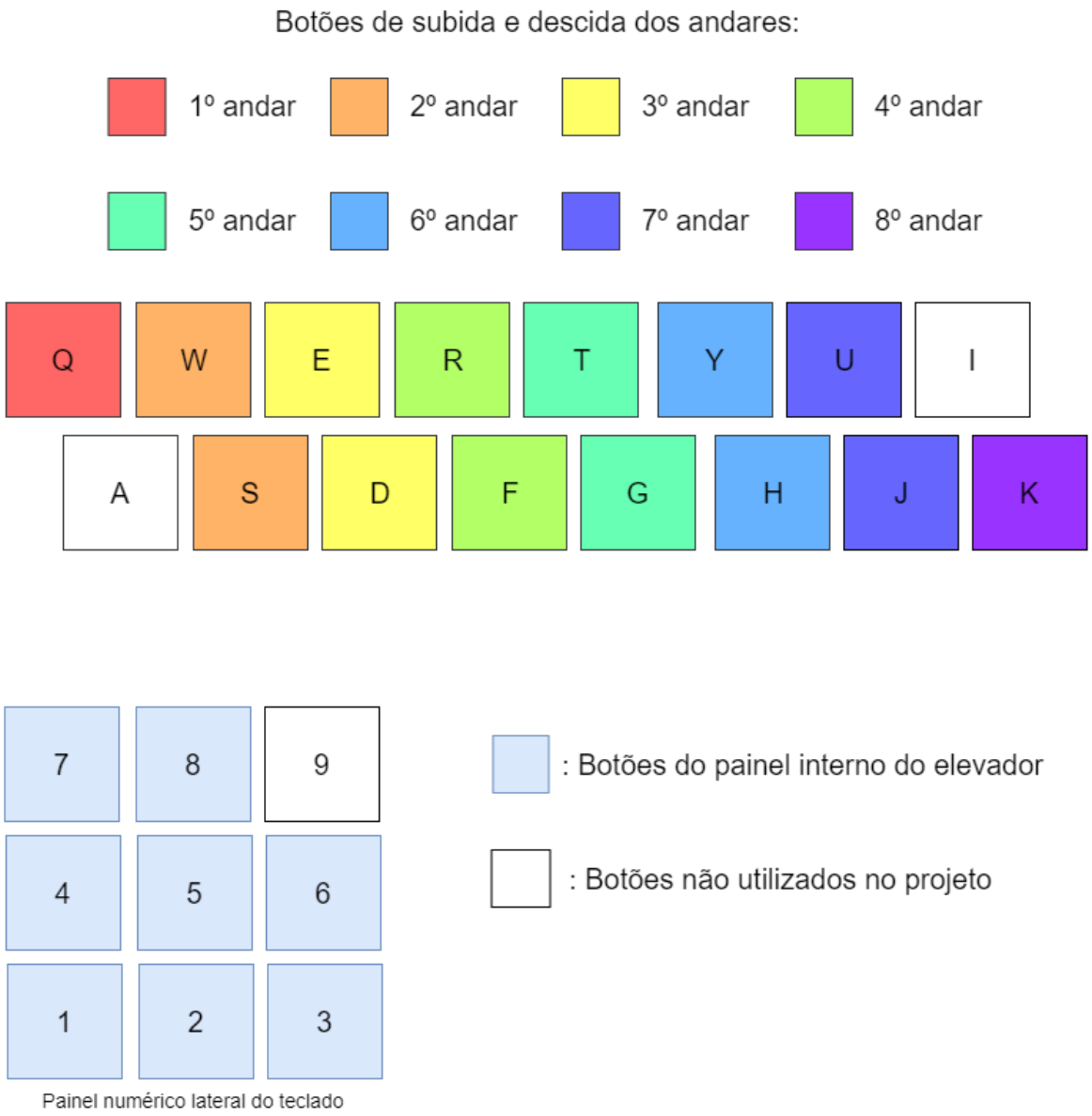

Figura 6 - Ilustração do mapeamento do teclado.

5. Relatório de compilação

Segue abaixo (Figura 7) o relatório de compilação do projeto na plataforma Quartus Prime. Como nosso projeto não faz uso de uma máquina de estados, maiores detalhes sobre o assunto não foram incluídos neste relatório.

Flow Summary

<<Filter>>

Flow Status	Successful - Tue Jun 19 20:24:20 2018
Quartus Prime Version	17.1.0 Build 590 10/25/2017 SJ Lite Edition
Revision Name	controlador_elevador_f12
Top-level Entity Name	controlador_geral
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	442 / 32,070 (1 %)
Total registers	427
Total pins	60 / 457 (13 %)
Total virtual pins	0
Total block memory bits	39,424 / 4,065,280 (< 1 %)
Total DSP Blocks	0 / 87 (0 %)
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	1 / 6 (17 %)
Total DLLs	0 / 4 (0 %)

Figura 7 - Relatório de Compilação no Quartus

6. Conclusão

O projeto desenvolvido cumpre de forma satisfatória as especificações definidas no início do desenvolvimento do projeto, e consegue simular bem o funcionamento de um elevador real. Ele também permitiu uma revisão de todos os conceitos abordados em aula durante o semestre e a utilização desses conceitos de maneira mais prática e aplicada à resolução de um problema real. Além disso, esse projeto foi realizado de forma modular, o que é uma boa prática a ser desenvolvida e treinada quando tratando-se de trabalhos em grupos.